REMARKS

In the Official Action dated April 19, 2006, the drawings were objected to for containing certain reference numerals not listed in the specification. Claims 1, 6-9, 11-13, 15-17, 20, and 26-29 were rejected under 35 USC § 112. Claims 11-30 were rejected under 35 USC § 101. Claims 1-4, 6-8, 10-14, 20-24, 26-28, and 30 were rejected under 35 USC § 102(e) as being anticipated by US PGPub 2005/00500000 (Kwok). Claims 5, 7-19, and 25 were rejected under 35 USC § 103 as being obvious over Kwok. Claims 9, 15, and 29 were rejected under 35 USC § 103 as being obvious over Kwok in view of US Pat. 6, 799,299 (Li). The various objections and rejections are addressed in turn below.

Objection to the Drawings

Regarding item 4, paragraph 0030 of the specification is amended to address the objection to the drawings. The language now refers to items 401a, 401b, and 401c as requested in the Official Action. Additional amendments to the specification have also been made where various minor matters of form were discovered, namely incorrect references to certain nonessential features of the drawings were discovered.

Rejections under 35 USC § 112

Regarding item 7, claim 1 is amended to confirm Examiner's view that the various references to an XSLT transform may be considered to refer to a same XSLT transform. After the "XSLT transform" limitation is introduced, subsequent references to it now read "said XSLT transform."

Regarding item 8, the term "object" has been substituted for the term "process" throughout the claims to address the stated confusion. This exchange is supported in the specification, for example by the reference to object-oriented programming in the summary and paragraph 0029. In object-oriented programming, one object may inherit another object, as described in the claim. When code associated with an object executes, it does so in a "process." While "process" was the term originally chosen for the claim, "object" may clear up confusion without significantly changing the scope of the claim.

Regarding item 9, because aspects of the invention are directed to creating variable features of XSLT transforms while reducing the work associated with creating non-variable features, the term "prefabricated" was selected to imply a custom transform that was pre-made for incorporation into an output transform (as stated in the summary). Because the term "custom transform" is a sufficient term to describe the invention and avoids the stated confusion, this term has been placed into the claims.

Regarding item 10, paragraph 0009 of the specification states that "an XSLT processor compares the elements in an input XML document to the templates in a stylesheet. When it finds a matching template, it writes the template's contents into an output or resultant tree." A template should therefore be understood to be a specific portion of an XSLT stylesheet that can be compared to portions of an input XML document. Applicants request that the term "template" be thus viewed as a specific portion of a stylesheet, rather than a stylesheet as a whole.

Regarding item 11, the language "abstract named template" is changed to "stub template" to resolve the confusion on the use of the term "abstract." As provided in the specification at paragraph 0045:

> The base transform 203 may then provide for calling an abstract named template, or stub template. The call to such an abstract named template can be overridden by the deriving transform 201. In short, the call to a stub template can initiate a process whereby the deriving transform 201 provides a "type handler" for a particular type of template. The type handler may provide additional features for an output transform 204 that are not provided by a base transform 203. As the name indicates, a type handler may be designed for use with particular types of data. For example, it can be tailored to a particular new file 104 format.

Regarding item 12, Applicants regard "readability" as similar in scope to "interoperability," and therefore have amended claim 12 to incorporate the Examiner's suggestion.

Regarding item 13, the concern is similar to that expressed in item 9, and Applicants have similarly replaced the term "prefabricated" with "custom."

Regarding item 14, the word "third" has been struck from the claim to avoid confusion surrounding use of the term "third component" in claim 18. Applicants do not

intend the components of claims 20 and 18 to be understood as the same component. The amendment provides the desired clarification.

Rejections under 35 USC § 101

With regard to item 16, claims 11- 16 have been amended so as to limit the claims to systems comprising computer readable media bearing data structures and/or instructions. Therefore, claims 11-16 should be treated as product claims under MPEP 2105. Claims 21-30 are amended to clearly indicate a computer comprising the recited means. Those of skill in the art will appreciate that virtually any computerized functions can be accomplished in either hardware or software. Therefore claims 21-30 should also be treated as product claims under MPEP 2105.

With regard to item 17, claims 17 -20 have been amended to recite a computer-readable medium bearing instructions for a computer. Applicants believe the amendments should satisfy the Examiner's concern.

Rejections under 35 USC § 102 and § 103

Claims 1-4, 6-8, 10-14, 20-24, 26-28, and 30 were rejected under 35 USC § 102(e) as being anticipated by US PGPub 2005/00500000 (Kwok). Claims 5, 7-19, and 25 were rejected under 35 USC § 103 as being obvious over Kwok. Claims 9, 15, and 29 were rejected under 35 USC § 103 as being obvious over Kwok in view of US Pat. 6, 799,299 (Li). The outstanding rejections under 35 USC § 102 and § 103 are respectfully traversed.

Both § 102 and § 103 require that the reference, or combination of references, disclose every element of the claimed invention. In this case, there is at least one element of each independent claim that is not found in the references. The various independent claims are addressed in turn below.

Claim 1 (as amended above) provides:

> 1. A method for automatically generating all or part of an Extensible Stylesheet Language Transforms ("XSLT") transform for transforming Extensible Markup Language ("XML") data in a source file format into data in a new file format, comprising:

producing an input file that identifies at least one data pattern from an XML source file; and

producing a first object for generating an XSLT transform, wherein said first object generates at least one first feature of said XSLT transform, and wherein said first object is designed to inherit a second object for generating said XSLT transform; and

incorporating said second object into said first object, wherein said second object uses said input file to generate at least one second feature of said XSLT transform.

As can be seen in the bolded text above, the claim recites, not producing a transform itself, but rather producing a first object that generates a feature of a transform. Furthermore, the first object inherits a second object for generating a transform.

In contrast, Kwok discloses creating content rules and presentation rules that are combined in a "combining operation" to produce a transform. See Kwok paragraph 0047. Kwok does not disclose that the presentation rules could be considered an "object" in the sense of object-oriented programming, nor does Kwok disclose that the "content rules" inherit the "presentation rules" or vice-versa.

The Official Action recognizes, for example in item 21, third paragraph, that object-oriented programming and inheritance approaches are not disclosed in Kwok or Li. However, the Official Action states that "it was well known in the art... that a process could produce a call that retrieves a stylesheet from memory." Official Action item 21, third paragraph. Applicant notes that this is an overly simplistic view of what occurs when a first object inherits a second object. Inheritance is more than just retrieving from memory. For example, as recited in Application paragraph 0049, "a deriving transform 201 may import a base transform 203." Application paragraph 0050 goes on to state that, "[t]he deriving transform 201 may also implement the abstract named templates, or stub templates, generated by the base transform 203."

Neither inheritance of a transform nor inheritance of an object for generating a transform is discussed in Kwok, Li, or the other references of record. Furthermore, Applicant submits that the inheritance-based approach provided in the claims is novel in this context and provides unique advantages of eliminating much tedium in transform generation. For example, in Kwok, the "content rules may be manually entered by an administrator directly only once for each new device or device type and each DTD or XML schema." Kwok

paragraph 0045, lines 13-15. Kwok also states, "it is contemplated that presentation rules may be manually entered by an administrator directly only once for each new device or device type." Kwok paragraph 0046, lines 5-7. Thus, Kwok contemplates combining two manually entered sets of rules (perhaps generated with some degree of automated assistance), without the benefit of inheritance approaches.

Furthermore, the Official Action mistakenly states in item 19, 3rd bullet that Kwok's generation of content rules according to the markup document is similar to the use of an input file to "generate at least one second feature of said XSLT transform" as recited in claim 1. This element of claim 1 refers to an *object* that uses the *input file* to generate an *output* – here, the output is a feature of a transform. In contrast, Kwok's content rules are *themselves produced* according to Kwok's markup document.

Claims 11 and 21 (as amended above) provide:

> 11. A system comprising a computer readable medium bearing data structures and instructions for generating XSLT transforms, comprising:
> a first data structure comprising an input file containing at least one XPath expression; and
> **a first XSLT transform comprising** instructions for generating at least one first feature of an output XSLT transform and **instructions for incorporating a second XSLT transform**, said second XSLT transform comprising instructions for generating at least one second feature of an output XSLT transform; and
> a object in said second XSLT transform, said object comprising instructions for generating an XSLT template or a portion thereof based on said at least one XPath expression.
>
> 21. A computer comprising means for automatically generating all or part of an Extensible Stylesheet Language Transforms ("XSLT") transform for transforming Extensible Markup Language ("XML") data in a source file format into data in a new file format, comprising:
> means for reading an input file that identifies at least one data pattern from an XML source file; and
> means for generating at least one first feature of an output XSLT transform with a first object; and
> **means for incorporating a second object for generating an XSLT transform into said first object**; and
> means for said second object to use said input file to generate at least one second feature of said output XSLT transform.

Claims 11 and 21 contain limitations that are similar to the requirement in claim 1 for producing a first object that inherits a second object. Claim 11 requires a first XSLT transform comprising instructions for incorporating a second XSLT transform. Claim 21 requires "means for incorporating a second object for generating an XSLT transform into said first object." Kwok is deficient for the same reasons as in the above discussion of claim 1.

Claim 17 (as amended above) provides:

> 17. A computer readable medium comprising computer readable instructions, said instructions comprising an XSLT transform, comprising:
> a first component comprising instructions for transforming at least one section of an input file into an XSLT template or portion thereof; and
> **a second component comprising instructions to call to a stub XSLT template.**

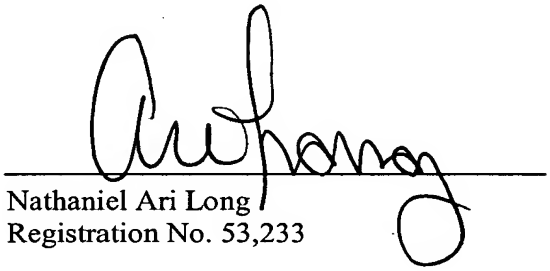
Here, the call to a stub template is a specific technique for implementing inheritance behaviors in the first component. The Official Action admits on page 15, first paragraph, that Kwok fails to disclose this. However, the Official Action alleges that, first, it would have been well known in the art that a process could produce a call that retrieves a stylesheet from memory, and second, that one of skill in the art would have found motivation to combine the elements of the claim.

Regarding the first assertion, it should be emphasized that the call to a stub template is not simply for retrieving a stylesheet from memory. As recited in Application paragraph 0045, "the call to an abstract named [stub] template can be overridden by the deriving transform 201. In short, the call to a stub template can initiate a process whereby the deriving transform 201 provides a "type handler" for a particular type of template." Thus, the call to a stub template is not so much to retrieve a stylesheet from memory as it is to pass control to the deriving transform.

Regarding the assertion that there would have been motivation to combine, Applicants request some further showing, if it can be found, that there would have been motivation to combine the specific elements of the claim.

Because independent claims 1, 7, 11, and 21 are not anticipated by Kwok, nor obvious over Kwok in view of Li and/or the other references of record, and because the remaining dependent claims incorporate the limitations of the underlying independent claims, Applicants submit that the claims in their present form are now allowable. Applicants respectfully await Examiner's further determination upon consideration of this response.

Date: July 19, 2006



Nathaniel Ari Long
Registration No. 53,233

Woodcock Washburn LLP
One Liberty Place - 46th Floor
Philadelphia PA 19103
Telephone: (215) 568-3100
Facsimile: (215) 568-3439